11-12-87

VZ - LINK

NEWSLETTER

» AUCKLAND VZ300/200 USERS CLUB «

ISSUE No:15 DECEMBER 1987

MERRY CHRISTMAS to ALL

Well here we are again , with the last issue of 1987.

What a funny year it has been on the VZ scene.D.S.E have released nothing new for the VZ and in fact seem to have given up on their so called " Little Wonder".

But luckly for us user's there are people out there that do care about the VZ and are not only willing to write software and item's for news letters,but have got organised and now market some of the best hardware and software available.

To these gentlemen we can only say a very big Thank you and please keep up the sterling work for us.

There is a lot of new software becoming available in Australia.This software is really great and is worth the investment.

The first issue for 1988 will be ready and posted out about late January and hopefully will include item's on machine code etc (as promised).I will also publish the second part of David Boyce's AEM 4505 SPEECH SYNTHESISER note's.

I hope you all enjoy this issue and hopefully you get time these hoilday's to write some item's for the newsletter and also enter some of the program's that have appeared in past issue's.

Please note that some of the printer interface's being sold at the moment have the early circuit board's and therefore you may have problem's getting "Quickwrite" which is the super wordprocessor program sold by John D'Alton ,to work with these unit's.I have spoken to John and there doe's not seem to be any easy answer to the problem.

Some of you may have seen my demo disk and club poster in the Dick Smith stores.What do you think?.
While on about D.S.E ,there are some new face's in most stores and we welcome these gentlemen

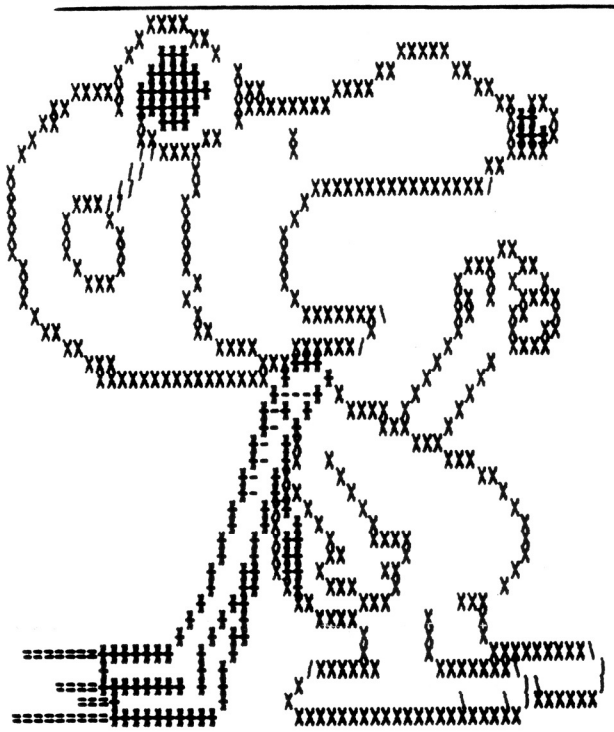
I would like to wish all user's and member's a MERRY CHRISTMAS and HAPPY NEW YEAR.

And now turn the page and enjoy the super content's of this issue .

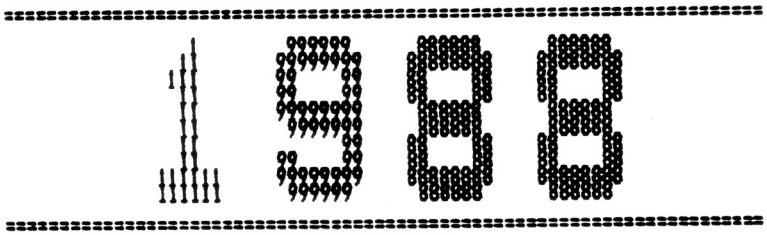
SILVER MOUNTAIN PT - 5

```
4510 G$="":FORI=1TO8
4520 F$=MID$(B$,1+INT(RND(0)*4)*3,1)
4530 G$(1)=G$(1)+F$
4540 IFF$="N"THENL$="S"
4550 IFF$="S"THENL$="N"
4560 IFF$="E"THENL$="W"
4570 IFF$="W"THENL$="E"
4580 G$(2)=L$+G$(2)
4590 NEXTI:RETURN
4670 OPEN"S.M.DATA",0:PRINT"  LOADING"
4680 FORI=1TO80:IN#"S.M.DATA",E$(I):NEXT
4690 FORI=1TO6:IN#"S.M.DATA",C(I):NEXT
4700 FORI=1TO70:IN#"S.M.DATA",F(I):NEXT
4710 IN#"S.M.DATA",G$(1),G$(2)
4720 CLOSE"S.M.DATA"
4730 R=F(69):R$="OK.CARRY ON":RETURN
4740 F(69)=R
4750 OPEN"S.M.DATA",1
4760 PRINT"  SAVING DATA"
4770 FORI=1TO80:PR#"S.M.DATA",E$(I):NEXT
4780 FORI=1TO6:PR#"S.M.DATA",C(I):NEXT
4790 FORI=1TO70:PR#"S.M.DATA",F(I):NEXT
4800 PR#"S.M.DATA",G$(1),G$(2)
4810 CLOSE"S.M.DATA"
4820 PRINT"  BYE.....":END
4830 LS=1:LP=1
4840 FORI=1TO LEN(J$)
4850 IFMID$(J$,I,1)=" "ANDLL>EL PRINTMID$(J$,LP,LS-LP):PCP=1
4855 IFPCP=1THENLL=I-LS:LP=LS+1:PCP=0
4860 IFMID$(J$,I,1)=" "THENLS=I
4870 LL=LL+1:NEXTI
4880 PRINTMID$(J$,LP,LEN(J$)-LP);
4890 RETURN
4900 EL=31:GOSUB4400:PRINT" DO YOU WANT TO"
4910 PRINT:PRINT"     1. SAVE ON CASSETTE"
4920 PRINT" OR 2. SAVE ON DISK"
4930 PRINT:PRINT" TYPE 1 OR 2 AND PRESS <RETURN>"
4940 INPUTC:IFC<1ORC>2THEN4930
4950 IFC=1THEN5150
4960 IFC=2THEN4740
4970 EL=31:GOSUB4400:PRINT" DO YOU WANT TO"
4980 PRINT:PRINT"     1. LOAD FROM CASSETTE"
4990 PRINT" OR 2. LOAD FROM DISK"
5000 PRINT:PRINT" TYPE 1 OR 2 AND PRESS <RETURN>"
5010 INPUTC:IFC<1ORC>2THEN5000
5020 IFC=1GOSUB5070:RETURN
5030 IFC=2GOSUB4670:RETURN
5070 INPUT" PRESS PLAY ON TAPE,THEN PRESS   <RETURN>";J$
5090 FORI=1TO80:INPUT#"D",E$(I):NEXT
5100 FORI=1TO6:INPUT#"D",C(I):NEXT
5110 FORI=1TO70:INPUT#"D",F(I):NEXT
5120 INPUT#"D",G$(1),G$(2)
5130 R=F(69):R$="OK.CARRY ON":RETURN
5150 F(69)=R
5160 INPUT"  PRESS'RECORD'AND'PLAY',THEN     PRESS <RETURN>";J$
5170 PRINT" OK. SAVING"
5180 FORI=1TO80:PRINT#"D",E$(I):NEXT
5190 FORI=1TO6:PRINT#"D",C(I):NEXT
5200 FORI=1TO70:PRINT#"D",F(I):NEXT
5210 PRINT#"D",G$(1),G$(2)
5220 PRINT" BYE.....":END
```

SNOOPY CALENDAR



BY DAVE BOYCE



1988

JANUARY

SU	MO	TU	WE	TH	FR	SA
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

FEBRUARY

SU	MO	TU	WE	TH	FR	SA
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29					

MARCH

SU	MO	TU	WE	TH	FR	SA
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

APRIL

SU	MO	TU	WE	TH	FR	SA
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

MAY

SU	MO	TU	WE	TH	FR	SA
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

JUNE

SU	MO	TU	WE	TH	FR	SA
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

JULY

SU	MO	TU	WE	TH	FR	SA
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

AUGUST

SU	MO	TU	WE	TH	FR	SA
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

SEPTEMBER

SU	MO	TU	WE	TH	FR	SA
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

OCTOBER

SU	MO	TU	WE	TH	FR	SA
30	31					1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

NOVEMBER

SU	MO	TU	WE	TH	FR	SA
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

DECEMBER

SU	MO	TU	WE	TH	FR	SA
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

```
90 '  SNOOPY CALENDAR FOR THE
100 ' BROTHER M-1009 OR EQUIV.
110 ' DOT MATRIX PRINTER.
120 ' FILE - SNOOPYCL
130 CLS:PRINT:PRINT"DO YOU HAVE YOUR PRINTER"
135 INPUT" TURNED ON <Y/N>":DF$
140 IF DF$="Y" THEN 155
145 PRINT:PRINT" OH WELL !! - SWITCH OFF AND    COME BACK WHEN":
150 PRINT" YOU HAVE - BYE!!":END
155 CLEAR800
162 VZ$="====================================="
165 DIM L(11),YR(11,6,4),X$(9,9)
170 DATA 31,28,31,30,31,30,31,31,30,31,30,31
172 '---X--------X-X--------X-X--------X-X--------X-X--------X-X--------X
175 DATA" 000000 ","00000000","00     00","00     00","00     00"
180 DATA"00     00","00     00","00     00","00     00","00000000"," 000000 "
185 DATA"    1    ","   11    ","  111    ","   11    ","   11    "
190 DATA"   11    ","   11    ","   11    ","   11    "," 111111 "," 111111 "
195 DATA" 222222 ","22222222","22     22","       22","  222222"
200 DATA" 222222 ","222      ","22       ","22222222","22222222"
205 DATA"33333333","33333333","      33 ","     33  ","   333   "
210 DATA"    333 ","     333","33    33","33333333"," 333333 "
215 DATA"     4  ","    44  ","   444  ","  4444  "," 44 44  "
220 DATA"44  44  ","44444444","44444444","    44  ","    44  "
225 DATA"55555555","55555555","55       ","55       ","5555555 "
230 DATA"55555555","      55","55     55","55555555"," 555555 "
235 DATA" 666666 ","66666666","66     66","66       ","6666666 "
240 DATA"66666666","66     66","66     66","66666666"," 666666 "
245 DATA"77777777","77777777","      77","      77","     77  "
250 DATA"    77  ","   77   ","   77   ","   77   ","   77   "
255 DATA" 888888 ","88888888","88     88","88     88"," 888888 "
260 DATA" 888888 ","88     88","88     88","88888888"," 888888 "
265 DATA" 999999 ","99999999","99     99","99     99","99999999"
270 DATA" 9999999","      99","99     99","99999999"," 999999 "
272 '---X--------X-X--------X-X--------X-X--------X-X--------X-X--------X
275 FOR I=0 TO 11:READ L(I):NEXT I
280 FOR I=0TO 9:FOR J=0TO 9:READ X$(I,J):NEXT J:NEXT I
285 '
290 GOTO 440
295 FOR M=0TO 11
298 PRINT".":
300 W=0:DT=1
305 YR(M,D,W)=DT:DT=DT+1:D=D+1
310 IF D>6 THEN D=0:W=W+1:IF W>4 THEN W=0
315 IF DT<L(M)+1 THEN 305
320 NEXT M:SOUND1,1
325 RETURN
330 Q$=RIGHT$(STR$(Y),4):I1=VAL(MID$(Q$,1,1))
335 I2=VAL(MID$(Q$,2,1)):I3=VAL(MID$(Q$,3,1))
340 I4=VAL(MID$(Q$,4,1))
345 LPRINT:LPRINT
348 LPRINT TAB(25);"      ";VZ$:VZ$:LPRINT
350 FOR I=0TO 9
352 '--X------------------X----------X-----X---------X
355 R$="                    "+X$(I1,I)+"     "+X$(I2,I)
360 R$=R$+"     "+X$(I3,I)+"     "+X$(I4,I)
365 LPRINT TAB(25);R$
370 NEXT I
375 LPRINT:LPRINT TAB(25);"      ";VZ$;VZ$:LPRINT CHR$(18)
380 RETURN
```

```
385 LPRINT A$
390 FOR W=0TO 4:B$=""
395 FOR I=0TO 2:B$=B$+"     "
400 FOR J=0TO 6
405 IF YR(M+I,J,W)=0 THEN C$="    " ELSE C$=STR$(YR(M+I,J,W))
410 IF LEN(C$)<3 THEN C$=" "+C$
415 B$=B$+C$
420 NEXT J:NEXT I
425 LPRINT B$
430 NEXT W
435 LPRINT:RETURN
440 CLS:PRINTTAB(6)" VZ 200 CALENDAR":PRINT
445 PRINT" THIS PROGRAM WILL GENERATE A   CALENDAR FOR ANY ";
450 PRINT"YEAR IN THE    RANGE 1901 - 1999. ALL YOU HAVE TO ";
455 PRINT"DO IS SPECIFY THE YEAR!"
460 A$="     SU MO TU WE TH FR SA":A$=A$+A$+A$
462 '---X-----------X-X-------X-X------------------X----------X
465 H1$="            "+"JANUARY"+"                    "+"FEBRUARY"
470 H1$=H1$+"                    "+"MARCH"
475 H2$="             "+"APRIL"+"                      "
480 H2$=H2$+"MAY"+"                          "+"JUNE"
485 H3$="               "+"JULY"+"                       "+"AUGUST"
490 H3$=H3$+"                    "+"SEPTEMBER"
495 H4$="            "+"OCTOBER"+"                    "+"NOVEMBER"
500 H4$=H4$+"                    "+"DECEMBER"
502 '---X-----------X-X-------X-X------------------X----------X
504 SOUND1,1
505 INPUT" FOR WHICH YEAR WOULD YOU LIKE   A CALENDAR ";Y
510 PRINT" PLEASE WAIT -- I'M INITIALISING MY DATA....."
515 IF Y<1901 OR Y>1999 THEN PRINT:PRINT"OUT OF RANGE":GOTO 505
520 IF 4*INT(Y/4)=Y THEN L(1)=29
525 I=Y-1901:J=INT(I/4):I=I-4*J+2
530 K=5*(J-7*INT(J/7))+I:D=K-7*INT(K/7)
535 GOSUB 295
538 SOUND 3,3:PRINT:INPUT"PRESS <RETURN> WHEN READY...";I
540 PRINT" PLEASE WAIT ...I WILL NOW"
545 PRINT" PRINT-OUT YOUR CALENDAR FOR"
550 PRINT" THE YEAR-:";Y
552 GOSUB 820'TO SNOOPY ROUTINE
555 GOSUB 330
558 LPRINT CHR$(27)+"2"
560 M=0:LPRINT H1$:GOSUB 385
565 M=3:LPRINT H2$:GOSUB 385
570 M=6:LPRINT H3$:GOSUB 385
575 M=9:LPRINT H4$:GOSUB 385
580 CLS:LPRINT
585 INPUT"ANOTHER YEAR ";YY$:IF YY$="Y" THEN RUN ELSE 590
588 END
590 LPRINT CHR$(27);"@"
600 CLEAR 50:CLS:PRINT"BYE-BYE":FOR LL=1TO 10:LPRINT:NEXT LL
810 END
820 LPRINT CHR$(27);"3";CHR$(18); ' LINES 18/216 APART
830 LPRINT CHR$(15); 'CONDENSED
840 LPRINT:RETURN
1500 ERA"SNOOPYCL"
1600 SAVE"SNOOPYCL":CLS:DIR
```

CONTINUED NEXT ISSUE

UNDERSTANDING YOUR VZ - -

PART 4 - BY ROBERT QUINN

Knowing the structure of basic lines you can easily estimate the number of bytes a line occupies in memory from a basic listing of the line, allowing five bytes for overhead (next line address, line number, end of line marker), and you can edit basic lines by POKEing new numbers to any RAM addresses you wish.

LINE is a utility routine which can be added to a basic program and used for line analysis. When run, LINE will ask for a line number. Enter the number of any line in the program, and LINE will hunt through memory for the line, then begin to display the address of each byte in that line, the byte itself and the position of the byte in the line. Start address of next line will be calculated from bytes 1 and 2; line number will be calculated from bytes 3 and 4; the text of the line will PEEK out from byte 5 onward. Press any character key to display each successive byte. If a byte has an address greater than 32767, then the negative PEEK/POKE address will be displayed also. A deluxe version of LINE is listed at the end of this months article which finds the required program line instantly -- if it is there to be found -- and decodes the text bytes of the line, displaying the corresponding characters or basic words.

PEEK/POKE editing of basic lines allows you to do many things that simply cannot be done with VZ basic editing. Suppose you want to change most of the PRINT words in a program to LPRINT words. I had occasion to do this in a Z80 DISASSEMBLER program which I had adapted to run on the VZ from the version by Geoff Lehrere (pub. in APC) designed for the TRS-80/SYSTEM 80 micro. I wanted a version which would output all the data to my printer instead of to the screen. Rather than list my way through all the program lines, changing hundreds of PRINT words manually, I devised a variation of the LINE routine to do the job for me.

Or you may want a routine to renumber all (or a subset of) the lines in a program. A routine to invers all the characters, or certain characters, within the quotes of all PRINT statements? Rename A$ as D$ throughout a program? These and a wide variety of tasks can be accomplished now that we understand basic lines.

PEEPO is a general routine which can be adapted for many of these tasks. When added to a program and RUN, PEEPO asks fo a START line number and an END line number, and then works its way through the program from START to END. In line 1 variable A becomes the start address of the next line, variable B becomes line number of the current line being processed. In line 3 variable N becomes the address of the first text byte -- byte 5 -- of the current line. Line 4 makes variable C equal to the byte with address N, and K takes a value less than zero if opening quotes are encountered in the line or a value of zero if closing quotes are encountered. When K is negative, C codes for a character in the text of a PRINT, LPRINT or REM statement and C greater than 127, codes for a graphic or inverse character. When K is zero, C greater than 127 will be a token for a basic word. Line 7 increments N, and PEEPO PEEKs at all the text bytes in the line until C is zero, indicating end of line. Lines 5 and 6 are reserved for the action -- whatever you want PEEPO to do as it makes its way through the program.

To use PEEPO to change all PRINT words between START and END to LPRINT words in some program, you would enter a line like this and RUN:

cont....->

```
5 IF k=0 AND C=178 THEN POKE n,175
```

178 is the token code for PRINT; 175 is the token code for LPRINT. PEEPO will carry out the conversion after you input start and end line numbers for the section of the program you wish to convert.

To renumber all the lines in the program, making the multiples of 10, change line 3 like this:

```
3 L=L+10:N%=L/256:N=L-N%*256:POKE N(2),N:POKE N(3),N%:GOTO 8
```

This routine will not recalculate GOTO or GOSUB numbers, and make sure that it does not renumber itself!

Perhaps you want to change a normal hash symbol appearing in numerous PRINT statements into an inverse hash?

```
5 IF K=-1 AND C=35 THEN POKE N,227
```

Convert A$ to D$?

```
5 E=N+1:IF E>32767 THEN E=E-d
6 IF K=0 AND C=65 AND PEEK(E)=36 THEN POKE N,68
```

These are a sample of the many possibilities of PEEK/POKE editing. You now have the information you need to set about designing your own routines to perform tasks to serve your needs. But what about the hidden facilities I mentioned at the start of this article? -- I thought you'd never ask! What are they? How are they realised?

Well, the operating system of the VZ contains machine code routines for processing basic statements of which the basic words were never implemented in the VZ. Running TOKEN (page 3 Issue number 13) you will have noticed that many of the numbers display without any corresponding basic words appearing on the screen. This is because the TABLE OF TOKENS (WORD TABLE) in the VZs ROM has had the basic words obliterated. Table 1 is a list of some token codes and their basic words which can be made to work on the VZ despite the decision by the powers that be to leave the out of the VZ. The 'syntax'is simply grammatically correct use of statements involving these words as recognised by the VZs operating system. In the list, an upper case X is used for a variable (numeric or string); lower case letters are used for constants -- substitute the numbers of your choice when using these statements. Most of the remaining blank token codes in the table of tokens are DISK BASIC tokens, none of which are implemented in VZ DOS BASIC.

Each of the new basic words can be implementedby reserving a place in memory for the token code and using the correct syntax for a basic statement involving that word. Since a token code requires just one byte of memory we need only reserve a single position (memory cell) for the code. A single character code (any charcter -- letter, digit, whatever) will do fine. But it would be handy to use a number that is instantly recognisable as reserved for this purpose when we PEEK through memory. The token code for the basic word LET (140) is ideal. LET is an utterly redundant basic word. Nobody uses it in basic -- do they? Of course not! So it is about timeLET paid its way -- no free residence in the VZ.

cont....->

To demonstrate the technique, we'll first try it out on a VZ basic word. Load the LINE routine into your VZ. Then enter this line:

```
10 LET"TEST":LIST 10
```

If you RUN you will get a SYNTAX ERROR message because the syntax is that of a PRINT statement. Since LET is the first text byte of the first basic line in program memory, we already know the address -- the fifth byte of the first line is in cell 31469, isn't it? If you would still like to see for yourself then RUN 50000 and input 10 as a line number. Now enter this command:

```
POKE 31469,178:RUN
```

So long as we choose the syntax to suit the basic statement we have in mind, we can place LET -- or whatevr character we choose -- in any basic line in our current program, RUN the LINE routine to identify the address of LET (or negative address if address is greater than 32767) and then POKE to that address the token code for the basic word we wish to substitute for LET.

Now we'll try it one of the new words.

```
10 PRINT LET(X),LET(X$)
```

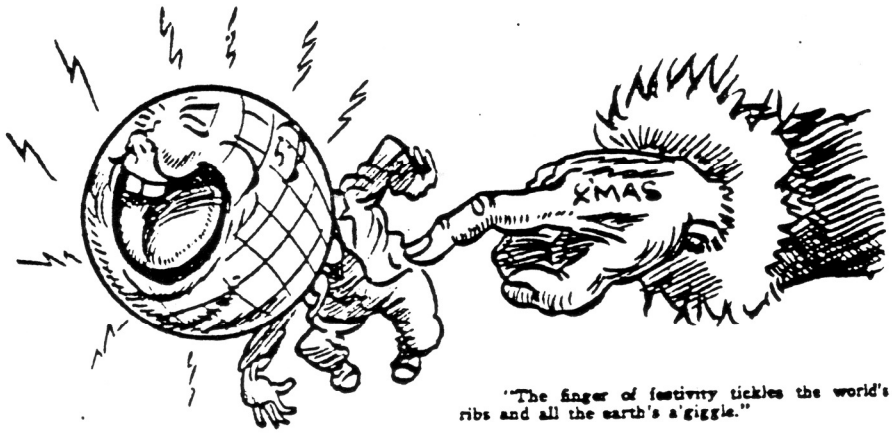
Now POKE 31470,218:POKE 31475,218

the two POKE addresses being for the two LETs and the 218 being the token code for FRE. This is the result:

```
10 PRINT FRE(X),FRE(X$)
```

But if you LIST 10 you will find nothing displayed of the text of line 10 beyond the first PRINT. This is typical for all basic words that have not been implemented in the VZ. When the listing routine encounters such a token code it has no way to display the basic word thereof and the rest of the line text is invisible. But it is all there in the program memory and can be PEEKed using the LINE routne.

You cannot basically edit any line which has been infiltrated with one or more of these foreign basic words. You must use PEEK/POKE editing. But other basic lines can be added to or deleted from your program and the lines occupied by the foreigners will shift their locations in memory accordingly without suffering any injury. Any good book on TRS-80 BASIC will give the ins and outs of these basic words. Next month I will concentrate on two or three that I consider especially useful.



"The finger of festivity tickles the world's ribs and all the earth's a'giggle."

TABLE 1.

TOKEN CODE	BASIC WORD	SYNTAX
218	FRE	PRINT FRE(X) displays amount of unused program memory PRINT FRE(X$) displays aount of free memory in string space
200	MEM	PRINT MEM same as FRE(X)
182	DELETE	DELETE a - b delete the lines a through to b from program
183	AUTO	AUTO a,b auto line numbering from a, increment b
220	POS	X=POS(X) get cursor (tab) position
161	on	ON X GOTO a,b,c,... computed GOTO ON X GOSUB a,b,c,... computed subroutine call
196	STRING$	X$=STRING$(a,"b") or PRINT STRING$(a,"b") explained in text **
192	VARPTR	X=VARPTR(X$) or Y=VARPTR(X) or X=PEEK(VARPTR(X$)) or POKE VARPTR(X$) explained in text **
153	DEFINT	DEFINTX or DEFINTX-Y define as interger variable
154	DEFSNG	DEFSNGX or DEFSNGX-Y as single precision variable
155	DEFDBL	DEFDBLX or DEFDBLX-Y as double precision variable
239	CINT	CINT(b) convert to interger: range -32768 to 32767
240	CSNG	CSNG(b) convert to single precision can use numeric
241	CDBL	CDBL(b) convert to double precision variables
242	FIX	FIXb or FIX(X) truncation: returns integer part
158	ERROR	used in ON ERROR GOTO
159	RESUME	RESUMEa program resumes running after error occurs
134	RANDOM	RANDOM reseeds the random number generator

NOTE: Use SOUND in place of ERROR eg. ON SOUND GOTO

** Text referred to is part 5,

P.O.BOX 1972 C.P.O AUCKLAND N.Z

YOU ARE WELCOME TO PHONE ME BETWEEN 6.30 PM AND 9 PM ----PHONE 673-967

DELUXE LINE

```
50000 POKE30862,39:POKE30863,114:M=65536:N=32767
50010 IFPEEK(29223)<>33ORPEEK(29224)<>233THENGOSUB50410
50030 INPUT"[illegible]";D$:IFD$=""THEND$="0"
50040 B=VAL(D$):IFASC(LEFT$(D$,1))>57,GOSUB50360:PRINT:GOTO50030
50045 PRINT:INPUT"TOKEN";E:IFE=9,E=140
50050 R=INT(B/256):POKE29221,B-R*256:POKE29222,R:R=USR(R)
50060 A=PEEK(29219)+PEEK(29220)*256:POKE30862,0:POKE30863,114
50070 FORR=0TO3:A(R)=A+R:IFA(R)>NTHENA(R)=A(R)-M:NEXTELSENEXT
50080 K=PEEK(A(0))+256*PEEK(A(1)):IFK=0THENPRINT"[illegible]";B:END
50090 D=PEEK(A(2))+256*PEEK(A(3))
50095 PRINT"[illegible]"
50100 FORR=ATOK-1:C=R:IFC>NTHENC=C-M
50110 F=F+1:Z=PEEK(C):PRINTR;F;TAB(11);Z;:IFF<5THEN50180
50120 IFZ=34THENP=NOTP
50130 IFP=-1THENPRINTTAB(17);CHR$(Z);:GOTO50180
50140 IFZ=0THENPRINT"[illegible]";:GOTO50180
50150 IFZ<95THENPRINTTAB(17);CHR$(Z);ELSEGOSUB50300
50180 IFR>NTHENPRINTTAB(24);C;
50200 IFF=2THENPRINT:PRINT"[illegible]";K
50210 IFF=4THENPRINT:PRINT"[illegible]";D
50220 PRINT:IFF<5THENNEXT
50230 IFE>127THENIFZ<>ETHEN50280ELSESOUND30,2;20,2
50240 F$=INKEY$:F$=INKEY$:IFF$=""THEN50240ELSEIFF$="6",RUN50000
50250 IFF$="0"THENINPUT"[illegible]";A:POKEA,140
50260 IFF$="X"THENINPUT"[illegible]";T:IFT=0,T=140:POKEC,TELSEPOKEC,T
50270 IFF$="L"THENINPUT"[illegible]";A:INPUT"TOKEN";T:POKEA,T
50280 SOUND0,1:NEXT:END
50300 IFZ=192THENPRINT"VARPTR";:RETURN
50320 POKE29261,Z:R=USR(R):R%=PEEK(29261)+256*PEEK(29262)
50330 PRINTCHR$(PEEK(R%)-128);:FORR%=R%+1TO6175:H=PEEK(R%)
50340 IFH>127,R%=6175ELSEIFH>0,PRINTCHR$(H);ELSEPRINTCHR$(223);
50350 NEXT:RETURN
50360 D=127:FORR=5712TO6175:B=PEEK(R):IFB>128THEND=D+1
50370 IFB>169THENA$=CHR$(B-128)
50380 IFB>31ANDB<96THENA$=A$+CHR$(B)
50390 IFA$=D$THENPRINTD:R=6175
50400 NEXT:RETURN
50410 DATA14,127,58,77,114,71,33,80,22,126,254,128,56,9,12,120
50420 DATA185,32,4,34,77,114,201,35,62,24,188,32,236,62,31,189
50430 DATA32,231,201,0,0,0,0
50440 DATA33,233,122,34,35,114,78,62,00,190,35,70,32,03,190,40
50450 DATA16,35,94,35,86,58,37,114,187,32,7,58,38,114,186,32,1
50460 DATA201,96,105,24,221
50470 FORR=29184TO29260:READA:POKER,A:NEXT:RETURN
```

PEEPO

```
0 A=31465:D=65536:DIMN(3):INPUT"[illegible]";F:INPUT"[illegible]";H:GOTO8
1 A=PEEK(N(0))+PEEK(N(1))*256:B=PEEK(N(2))+PEEK(N(3))*256
2 IFA=0ORB>HTHENENDELSEPRINTB:N=N(0)+4:IFB<FTHEN8
3 IFN>32767THENN=N-D
4 C=PEEK(N):IFC=0THEN8ELSEIFC=34THENK=NOTK
5 :
6 :
7 N=N+1:GOTO3
8 K=0:FORR=0TO3:N(R)=A+R:IFN(R)>32767THENN(R)=N(R)-D
9 NEXT:GOTO1
```

DOT MATRIX PRINTERS

by Larry Taylor

The VZ is a versatile, little computer, especially if teamed with a peripheral device. When that device is an EPSON type dot matrix printer, its capabilities can greatly enhance the VZ's versatility.

Dot matrix printers form their characters from a series of dots. This type of printer falls into one of two categories either impact or non-impact. There are three main types of non-impact printers, thermal, electrostatic and ink jet. Thermal print heads contain a set of fixed pins, which are heated rapidly to produce a dot on specially treated, heat sensitive paper. Electrostatic print heads, also consist of fixed pins, which produce a tiny electrical discharge between the pins and an aluminium coated paper, forming the dot. Ink jet print heads are comprised of a vertical line of pin holes, through which a fine jet of ink is squirted onto the paper to create each dot.

Currently, impact type printers are the most common. The print head in an impact type printer usually consists of a line of pins arranged vertically. These pins strike against an inked ribbon to leave an impression on the paper. Each pin can be fired independantly. By firing all or some of these pins, a column of dots can be formed. The matrix used for most upper case (capital) letters is 5 dots wide and 7 dots high. To produce a character, the pins are fired forming a column of dots. The print head then moves one dot position and the pins fire again. This happens three more times to produce an upper case letter 5 dots wide.

Unfortunately, while the principle of their operation is similar, dot matrix printers differ in other ways. Software codes that are sent from the computer to the printer determine which pins will be fired. Not all printers will react to those codes in the same way. There are basically three families, EPSON, TANDY and APPLE, each conforming to a different standard. Both the EPSON and APPLE types address 8 pin print heads, whilst the TANDY types possess only 7 pins. Each pin is given a weighted value. In the case of EPSON types, the topmost pin has a value of 128, whilst the bottom pin has the value 1. These values are reversed for APPLE and TANDY printers. The bottom pin on a TANDY type is the seventh one and it has the value 64. TANDY type printers will only accept codes in the range from 1 to 127. The GP-100 belongs to the TANDY family and the VZ's ROM is set up to work with this type of printer. Since the codes used aren't compatible with EPSON type printers, a software patch is needed. The value of such a patch can be determined by looking at the relative merits of the two types of printers.

The GP-100 has tractor feed only, which requires the use of paper with sprocket holes. It has a print speed of 30 characters per second (30 cps) and is capable of unidirectional (left to right) printing only. This printer has both character and graphics modes. In character mode it possesses just the one typeface, which has no true descenders. That is, the tails on the lower case letters g,j,p,q and y do not descend below the line as they normally would. The only typestyle change, that can be made to the existing typeface, is that each character can be printed in double width (expanded) form.

On the other hand, software such as the VZ-EPSON Printer Patch, WORDPRO and QUICKWRITE, allows the VZ to make the most of the facilities provided by EPSON type printers. All allow EPSON codes to be sent directly to the printer.

Consequently, instead of being limited to the GP-100's meagre repertoire, a whole range of printers, with a host of enhanced features, is on offer. Most provide tractor and friction paper feeds, though a few require that the tractor feed be bought separately. In some cases, it may be possible to purchase a cut paper feeder, which can be useful, when printing multiple copies of correspondence on single sheets. Logic seeking (where the head looks to move the shortest distance possible) and bidirectional printing (the head prints on both left to right and right to left passes), means speeds can range from a minimum of 80 cps to 200 cps and more. These speeds are, of course, optimum values.

Though most printers possess only two typefaces (character sets), pica and italic, which are stored in an internal ROM, they compensate for this by offering a variety of typestyles. Different typestyles are formed by slightly modifying each character as it is printed. These include elite, condensed and double width. A few printers also offer double height and reverse print. In addition, there are features such as subscript (below the line) and superscript (above the line), emphasized and double strike. Proportional spacing is also available, which adjusts the distance between letters, according to their width. Some printers may have a high density (near letter quality) facility as well. With this the print head makes a second pass along each line, filling in the gaps between the dots, which form each character. Although slower, it results in better formed characters.

The ability to control line feeds to within 1/216 of an inch is desirable when compared to the GP-100's 1/6 of an inch. This is particularly useful to anyone, who has struggled to get mailing labels to line up on the printer. Some even allow the paper to be fed in reverse as well as forward. (This entire article was written using WORDPRO and printed in a single pass. The two column format was achieved by altering margin settings and using the printer's reverse feed.) The standard features all have recognized EPSON codes, whereas other features and their codes may vary from one printer to the next. The following table displays the codes, in decimal form, along with the resulting print type.

Standard Epson Codes

[27,80]	Normal Pica Print
[27,112,1]	**Proportional Print**
[27,77]	Elite Print
[27,52]	*Italic Print*
[14]	Double Width
[15]	Condensed Print
[27,69]	**Emphasized Print**
[27,71]	Double Strike Print
[27,45,1]	Underlined Print
[27,83,0]	Superscript Print
[27,83,1]	Subscript Print

Non-Standard Codes

[27,97,1]	High Density Print
[27,126,50,1]	REVERSED PRINT
[27,104]	DOUBLE HEIGH

Whilst there is considerable standardization of EPSON print codes, there seems to be very little when it comes to the format used to present them. When examining printer manuals, the presentation of these codes can appear quite confusing. Some examples are given below. Each is supposed to show which codes are needed to switch on Emphasized Print. As shown in the table, this simply involves sending the two values 27 and 69 to the printer.

Printer Manual 1 ESC+E [(45, C5)H, [69,197] D]
Printer Manual 2 ESC E
Printer Manual 3 <ESC> 69 45
Printer Manual 4 <ESC> E
Printer Manual 5 [1B]H [45]H or $[27]_{10}$ $[69]_{10}$

If you're confused by the above, don't feel bad, I found it just as difficult to understand? Basically the codes are presented in three ways, ASCII, Decimal (Base 10) and Hexidecimal (Base 16). ESC stands for Escape and represents the Decimal value 27. In Hexidecimal form the value 27 is equal to 1B. If a number has no identifying character or is followed by either a D or 10, then it is considered to be a decimal number. If it is followed by an H or 16, then it is considered to be Hexidecimal. So the three ways of representing the code required to switch an EPSON printer to Emphasized mode are summarised below.

ASCII	ESC	E
Decimal	27	69
Hexidecimal	1B	45

In VZ BASIC it would be accomplished by:

```
LPRINT CHR$(27);"E";
```
or
```
LPRINT CHR$(27);CHR$(69);
```

Word processing software such as QUICKWRITE and WORDPRO allow print codes to be embedded in the text. Since these programs bypass the VZ's printer driver, any code value used will be sent directly to the printer. The following examples, illustrate how each of these programs enable a change of typeface at the beginning of a line.

QUICKWRITE embeds codes at the start of a line of text. The code must be preceded by a carriage return <CR>.

eg. <CR>[27][52]This is a word processing program.

WORDPRO uses printer control lines which are preceded by a print flag <PF> and end in a carriage return <CR>.

eg. <PF>N=27,52<CR>
This is a word processing program.

The result, when printed from either program is as follows:

This is a word processing program.

Although the manual says otherwise, WORDPRO is able to change typefaces in the middle of a line. The procedure is a little more complicated and involves turning the line feed command on and off. My thanks to John Chapman for supplying the method used.

eg. <PF>F=N<CR> (set line feed off)
This is a<CR>
<PF>N=27,52<CR> (switch italics on)
word processing<CR>
<PF>N=27,53 F=Y<CR> (italics off, line feed on)
program.

The result, when printed is as follows:

This is a *word processing* program.

I've been informed that a recently updated version of QUICKWRITE will also be able to do this.

Whilst both of these approaches are satisfactory, they require the user to be familiar with all of the different print codes. Some word processing packages allow a label to be assigned to each print code, when the program is first set up. These can then be inserted in the text as required without the user having to know the code.

One problem often experienced, when attempting to use the VZ with an EPSON type printer, is that EPSON codes often alternate the values 1 and 0, for switching on and off a particular feature. For instance, the code used to enable and disable underlining [27,45,n] is identical, except for the last value n. Using the value n=1 selects underlining, whilst the value n=0 turns it off. The current VZ printer routine will not allow the value 0 through to the printer. Therefore, switching on underlining will work, but switching it off will not, and the same applies to similarly structured codes. Also affected, is any print code value greater than 127, which will be interpreted by the VZ's ROM routine as being either a graphics or inverse character. Since TANDY type printers do not store any characters in this range, the value will be intercepted by the VZ's ROM and replaced by a stream of graphics data. Whilst this data will enable a printer such as the GP-100, to print the character, it will not be compatible with an EPSON printer. These difficulties can be overcome by using a printer patch or by sending the value directly out the printer ports (0DH and 0EH). Print codes sent to the printer in this way, will be acted on without interference. It should be noted, however, that any characters printed by an EPSON printer, corresponding to codes in the range from 95 to 255, will not be the same as those displayed on the VZ's screen.

A number of printers come equipped with a RAM buffer. Instead of the computer being tied up waiting for the printer, the buffer acts as a storage area for data to be printed and so frees the computer sooner. A second use, is as an area for downloadable characters. These are shapes, which the user may design or which are modifications of the existing character set. By storing these shapes in the buffer, the printer can print them, as if they were part of the normal set.

In bit image (graphics) mode, print resolutions ranging from 480 to 1920 dots per line are possible. This mode allows the printing of shapes not held in the printer's ROM and is used when doing a dump of a HIRES screen. It should be noted that the VZ's screen is only 128 pixels wide. The printer is capable of 15 times this resolution, allowing it to depict much finer detail than is possible on screen.

In the world of printers, the primary standard has been set by EPSON. There can be no doubt of this, since the majority of printer and computer manufacturers produce equipment, which conforms to it. IBM's adoption of the standard is further confirmation. Even the TANDY printers, offer, in their latest models, an IBM-EPSON compatible mode. Your VZ may not be able to compete in the real world, but your printer can. Selecting an EPSON type printer to partner your VZ, means stepping into the mainstream of the computing world. When you finally outgrow your VZ, you will still have a useful peripheral that's able to communicate with almost any computer you happen to choose.